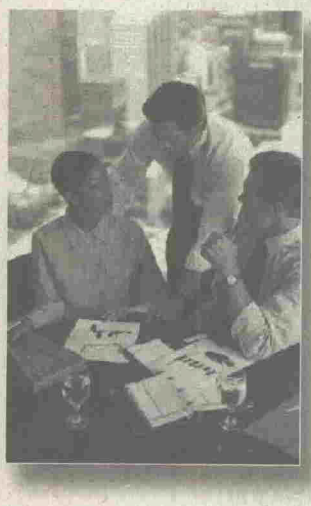
■ ยืน ภู่วรวรรณ

ปัญหาที่จะเกิดขึ้นกับซอฟต์แวร์ในอนาคต

ซอฟต์แวร์เป็นสิ่งที่มีการพัฒนาให้ใช้ประยุกต์ในงานด้านต่าง ๆ มากมาย ซอฟต์แวร์เป็นสิ่งที่ควบคู่กับการพัฒนาคอมพิวเตอร์ กว่า 50 ปีที่มีคอมพิวเตอร์ใช้งาน ซอฟต์แวร์ได้รับการเขียนและใช้งานกันหลายร้อยล้านการประยุกต์ มีผู้คนเขียนซอฟต์แวร์สำหรับคอมพิวเตอร์ตั้งแต่งานง่าย ๆ ที่มีการคำนวณไม่มาก จนถึงงานการใช้ในองค์กรที่มีความสลับซับซ้อนมาก ปัจจุบันซอฟต์แวร์ที่ใช้อยู่บางงานมีความสลับซับซ้อนเทียบเท่ากับการพัฒนาของคนกว่า 100 ปี อย่างไรก็ดี วงจรของซอฟต์แวร์ก็เป็นไปตามของสังขาร คือมีการพัฒนาขึ้นใหม่เรื่อย ๆ และก็มีที่เลิกใช้ไปแล้วเป็นจำนวนมากเช่นเดียวกัน หากจะถามว่าซอฟต์แวร์ที่ใช้อยู่นี้มีช่วงอายุยาวนานเพียงไร คงไม่มีใครสามารถตอบได้ เพราะบางส่วนของซอฟต์แวร์ยังคงมีความเกี่ยวโยงและนำกลับมาใช้หรือใช้เชื่อมโยงถึงได้

การพัฒนาซอฟต์แวร์ต้องมีวิสัยทัศน์ในการมองไปข้างหน้าที่ยาวไกล และต้องไม่สร้างปัญหาให้เกิดขึ้นในอนาคต



หากพิจารณาที่โลบรารีของภาษาที่ใช้ เช่น ภาษาซี ภาษาฟอร์แทน หรือปาสคาล โคบอลก็ดี โปรแกรมในโลบรารีอาจมีการพัฒนามานานแล้ว แต่ยังคงเก็บไว้ใช้ได้ ดังนั้นจึงมีซอฟต์แวร์เก่าที่เชื่อมโยงอยู่ด้วยเป็นจำนวนมากแน่นอน

ปัญหาของซอฟต์แวร์อยู่ที่การเปลี่ยนเงื่อนไขจากเดิม

ลองนึกดูว่าเมื่อพัฒนาซอฟต์แวร์ชุดหนึ่งสำเร็จและนำมาใช้งาน หากต้องมีการเปลี่ยนแปลงบางอย่างซึ่งเป็นการเปลี่ยนแปลงที่มีผลต่อโปรแกรม โดยที่โปรแกรมเหล่านั้นไม่มีการออกแบบให้รองรับการเปลี่ยนแปลงไว้ก่อน เช่น ถ้าต้องเปลี่ยนรหัสบุคลากรจาก 5 หลัก เป็น 6 หลัก เปลี่ยนหมายเลขโทรศัพท์ที่จากเดิม 7 หลัก เป็น 8 หลัก เหล่านี้ย่อมทำให้เกิดข้อยุ่งยากต่าง ๆ ตามมาเป็นจำนวนมาก

มองเตรียมการไปข้างหน้า

ผู้เขียนเคยทำงานระบบสารสนเทศให้มหาวิทยาลัยในหลายเรื่อง ต้องได้รับความยุ่งยากในการเปลี่ยนแปลงหลายครั้งที่มีผลต่อซอฟต์แวร์ที่เคยพัฒนา เช่น เปลี่ยนรหัสวิชาจากที่ใช้ตัวอักษรผสมตัวเลข (EEIII, ME305) มาเป็นตัวเลขล้วน ๆ (205111 206305) หรือเปลี่ยนรหัสบุคลากรกับเลขประจำตำแหน่งให้เป็นรหัสมาตรฐานขององค์กรที่ทุกหน่วยงานใช้ร่วมกัน การเปลี่ยนแปลงเหล่านี้มีผลกระทบต่อฐานข้อมูลเดิม ต่อรายงาน ต่อเอกสาร สิ่งพิมพ์ต่าง ๆ รวมทั้งการคุ้นเคยในการทำงานเดิม

แต่เหตุที่ต้องเปลี่ยนเพราะต้องมองไกลไปข้างหน้า การทำงานใดต้องมีมาตรฐาน และมีสภาพการใช้งานที่ไม่สร้างปัญหาในอนาคต มีความเหมาะสมกับการใช้งานในระบบสารสนเทศ ดังนั้นการเปลี่ยนแปลงรูปแบบมาตรฐานข้อมูลโดยคิดคำนึงถึงสภาพการใช้งานระยะยาวจึงเป็นเรื่องสำคัญ และทำให้องค์กรมีมาตรฐานใช้งานได้ดีต่อไป แต่จะต้องได้รับความยุ่งยากในการปรับแก้ไขระยะแรกอย่างมาก

ดังนั้นการพัฒนาซอฟต์แวร์ในปัจจุบันต้องมีวิสัยทัศน์ในการมองไปข้างหน้าที่ยาวไกล ต้องไม่สร้างปัญหาให้เกิดขึ้นในอนาคต ต้องวางมาตรฐานที่ใช้งานจากวันนี้และใช้ต่อไปในอนาคตได้ยาวไกล ผู้พัฒนาซอฟต์แวร์อาจต้องเปลี่ยนแปลงรูปแบบสภาพการทำงานที่เคยทำอยู่เดิม เพื่อจะได้ไม่มีปัญหาในอนาคต ดังตัวอย่างเช่น หน่วยงานรัฐบาลหลายหน่วยงานได้กำหนดรหัสกันขึ้นมาเองอย่างไม่คำนึงถึงสภาพมาตรฐานที่ใช้ เช่น รหัสประจำตัวผู้เสียภาษี รหัสคนไข้ รหัสประกันสังคม รหัสประจำตัวบัตรประชาชน รวมถึงรหัสอื่น ๆ อีกหลายอย่าง ทำให้ขาดความเป็นมาตรฐานกลาง หรือแม้แต่การกำหนดรหัสพื้นที่จังหวัด อำเภอ ตำบล ก็มีการกำหนดกันหลากหลายมาตรฐาน จนหากอนาคตมีการเปลี่ยนแปลงจะมีข้อยุ่งยากตามมา

การเริ่มต้นพัฒนาระบบงานหรือซอฟต์แวร์จึงต้องคำนึงถึงในเรื่องมาตรฐานเหล่านี้เป็นสำคัญ อย่างไรก็ตาม เมื่อเราคำนึงถึงสิ่งเหล่านี้อย่างดีแล้ว บางครั้งก็ยังกระทบกับปัญหาที่มา

จากภายนอก หรือสิ่งที่อาจเกิดขึ้นได้ในอนาคต โดยที่เราไม่คาดคิดมาก่อน

ตัวอย่างปัญหาที่จะเกิดขึ้น

หากมองไปข้างหน้าในเรื่องปัญหาต่าง ๆ ของซอฟต์แวร์ที่ทั่วโลกจะกระทบและเป็นตัวอย่างที่เราจะได้ศึกษาเพื่อหาทางป้องกันก่อน ปัญหาที่จะเกิดขึ้นและมีผลกระทบต่อซอฟต์แวร์ที่เด่นชัดมีมากมายหลายปัญหาที่ต้องพิจารณาและหาทางป้องกัน

ปัญหาเงินยูโร

สหภาพการเงินยุโรปได้สร้างแผนการรวมเงินให้เหลือสกุลเดียว เพื่อความแข็งแกร่งของชาติยุโรป เพราะจากอดีตแม้แต่เงินปอนด์ของอังกฤษก็เคยมีผลกระทบจากการโจมตีของนักค้าเงินระดับโลก ผู้หวังเก็งกำไรกันมาแล้ว การรวมเงินสกุลเดียวของยุโรปจะเริ่มตั้งแต่ 1 มกราคม 2542 เป็นต้นไป และจะค่อย ๆ ดำเนินการไปทีละประเทศจนเสร็จสิ้นในปี พ.ศ. 2545 (2002)

ปัญหานี้เป็นปัญหาใหญ่ที่ประเทศทางยุโรปจะได้รับผลกระทบอย่างหนัก ส่วนประเทศอื่น ๆ ก็จะได้รับผลกระทบตามด้วย เพราะจะต้องแก้ไขอัตราแลกเปลี่ยน ทั้งนี้การเปลี่ยนแปลงก็ยังไม่เปลี่ยนแปลงทันทีแบบเบ็ดเสร็จแต่จะค่อย ๆ เปลี่ยนแปลง การให้ซอฟต์แวร์รับรู้เงินตราสองสกุลพร้อมกันและมีเงื่อนไขพิเศษต่าง ๆ อยู่ในซอฟต์แวร์ตลอดช่วงเวลาของการเปลี่ยนแปลงเป็นเรื่องที่ยุ่งยาก

โปรแกรมประยุกต์เก่า รวมถึงข้อมูลที่บริษัทห้างร้านหรือองค์กรต่าง ๆ ใช้อยู่เป็นสกุลเงินเดิม แต่เมื่อเปลี่ยนมาใช้สกุลเงินใหม่ ความสับสนของการคำนวณและการดำเนินการในซอฟต์แวร์ย่อมต้องเกิดขึ้น ดังนั้นตลอดช่วงปีที่มีการเปลี่ยนแปลงและหลังการเปลี่ยนแปลงความสับสนของการใช้ซอฟต์แวร์จะมีมาก จำเป็นต้องแก้ไขซอฟต์แวร์ให้รับรู้และคำนวณเงินได้ถูกต้อง กล่าวกันว่าโปรแกรมที่จะต้องแก้ไขมีมากกว่าโปรแกรมปี 2000 อีก เพราะประเทศเหล่านี้ต้องใช้งานเกี่ยวกับเงินทองของตนเองอยู่แล้ว การแปลงโปรแกรมในประเทศยุโรปและประเทศอื่นที่มีความสัมพันธ์และใช้เงินร่วมกันจึงต้องเร่งกระทำและมีขั้นตอนตรวจสอบที่รัดกุม

ปัญหาปี 2000

เป็นปัญหาที่มีการกล่าวถึงกันมากแล้ว ปัญหาเกิดจากการใช้ปี ค.ศ. สองหลัก ซึ่งทำให้ค่าปี ค.ศ. 2000 กลับมาเริ่มต้นและสร้างความสับสนกับปี ค.ศ. 1900 ปัญหานี้เป็นปัญหาระดับโลกที่มีการดำเนินการกันมาแล้วและมีความตื่นตัวกันอย่างมากพอควร

ปัญหาการครบรอบของ GPS

ระบบ GPS ที่ใช้กันทั่วโลกจากดาวเทียม 24 ดวงที่ดำเนินการอยู่ในขณะนี้ ใช้งานกันอย่างแพร่หลายและกระจัดกระจายอยู่ทั่วโลก การเริ่มใช้งานถือกำหนดจากวันที่ 5 มกราคม 1980 การบันทึกสัปดาห์ที่อ้างอิงใช้หลักการของโมดูล 10 บิต ซึ่งมีค่าสูงสุดได้ 1,024 สัปดาห์ นั่นหมายถึงการครบกำหนดในวันที่ 21 สิงหาคม 1999 และสัปดาห์ของตัวมันจะกลับมาเป็น 0000 ใหม่อีกครั้ง หรือหากจะดูจากปัญหาจะเห็นว่าช่วงเวลาหนึ่งรอบที่กำหนดไว้คือประมาณ 20 ปี

อย่างไรก็ดี การแก้ไขระบบ GPS ได้กระทำจากระบบควบคุมดาวเทียมภาคพื้นดินที่ส่งสัญญาณเพิ่มเติมบางอย่าง เพื่อให้มีการรับรู้ในเรื่องวันที่ แต่โปรแกรมเก่าหรือผู้เขียนโปรแกรมที่ใช้กับ GPS และยังคงใช้งานเดิมอยู่จะไม่ทราบได้ หากไม่มีการแก้ไขโปรแกรมเหล่านั้น

หากพิจารณาให้ดีว่าทำไมเวลาของ GPS จึงมีความสำคัญ ทั้งนี้เพราะการดำเนินการใด ๆ บนโลกจำเป็นต้องมีการอ้างอิงเวลามาตรฐาน ซึ่งโดยปกติใช้เวลา UTC–Coordinated Universal Time เพื่อใช้ในการอ้างอิง เวลาของ GPS กับเวลา UTC จำเป็นต้องมีการปรับให้เข้ากันได้เสมอ และใช้งานในลักษณะที่จะอ้างอิงกันได้ทั่วโลก

ระบบเวลาที่อ้างอิงกันได้ทั่วโลกจึงเป็นระบบที่น่าจะเป็นประโยชน์ เพราะสามารถนำเวลามาใช้ประโยชน์ในลักษณะที่เป็นเวลาเดียวกันทั่วโลก

ปัญหาเลขเก้า

ปัญหาที่เกิดจากผู้เขียนโปรแกรมที่ใช้เลขเก้าในฟิลด์ข้อมูลต่าง ๆ เพื่อแทนค่าสูงสุดที่ฟิลด์นั้นจะรับได้ หรือใช้เลขเก้าเพื่อใช้เป็นรหัสจบไฟล์ ดังนั้นเมื่อใช้กับฟิลด์วันที่ การแทนเลขเก้าหมด จะหมายถึงวันที่ 9 เดือนกันยายน 1999 ซึ่งอาจแปลความหมายผิดจากที่ต้องการ เช่น ต้องการใช้เป็นเวลาหมดอายุที่ยาวนานจนเสมือนอินฟินิตี้ แต่ไม่สามารถใส่คำว่าอินฟินิตี้ได้ จึงใช้ค่ามากสุด แต่ปัจจุบันค่ามากสุดที่ใช้กำลังจะถึงจึงสร้างปัญหา

ในระบบยูนิกซ์ ถ้าใช้ตัวเลข 999999999 จะมีความหมายถึงการใช้บอกว่าจบไฟล์ แต่ถ้านำรหัสตัวเลขเก้าเก้าหลักนี้มาใช้ในการแปลความหมายเป็นวันที่ จะหมายถึงวันที่ 8 เดือนกันยายน ค.ศ. 2001 ซึ่งไม่ใช่รหัสสุดท้ายที่วันที่ในยูนิกซ์รันได้ ซึ่งเป็นปี ค.ศ. 2038

การใช้ตัวเลขสูงสุดในรีจิสเตอร์ที่นำมาแปลความหมายจึงขึ้นอยู่กับค่าของรีจิสเตอร์นั้นว่าใช้แทนอะไร ขอบเขตของรีจิสเตอร์เป็นขอบเขตที่มีความจำกัดใน

ตัวเอง ค่าของรีจิสเตอร์ที่ใช้มักเป็น 16 บิต 32 บิต ซึ่งถ้าแปลมาเป็นปริมาณก็พบว่ามีค่าไม่ใหญ่มากนัก ซึ่งแต่ก่อนค่าเหล่านี้อาจจะมากพอ แต่กาลเวลาผ่านไป เทคโนโลยีพัฒนาไปมาก ค่ารีจิสเตอร์ที่ใช้จึงเล็กลงไปมาก

ปีอธิกสุรทิน

ปีอธิกสุรทินเป็นอีกเรื่องหนึ่งที่มีการใช้งานผิดพลาดได้ หลายคนอาจคิดว่าปี ค.ศ. ที่สามารถหาร 4 ได้ลงตัวเป็นปีอธิกสุรทิน ถ้าเป็นเช่นนี้หมายถึง โลกหมุนรอบดวงอาทิตย์หนึ่งรอบใช้เวลา 365.25 พอดี แต่ในความเป็นจริงโลกหมุนรอบดวงอาทิตย์หนึ่งรอบไม่พอดีกับ 365.25 แต่มีเศษเกินอีกเล็กน้อย ซึ่งจำเป็นต้องสร้างวันที่แทรกไปในบางปี ดังนั้นเดือนกุมภาพันธ์จึงเป็นเดือนที่ใช้ปรับ

จากหลักการทางสุริยคติ ใช้ปีที่หาร 4 ลงตัว คงมีกุมภาพันธ์ 29 วัน แต่มีข้อยกเว้นว่า หากหารด้วย 100 ลงตัว ก็ให้มีเพียง 28 วัน และยกเว้นต่ออีกว่า ถ้าหาร 400 ลงตัว ก็ให้มี 29 วันอีก

ดังนั้นในปี ค.ศ. 2000 จึงเป็นปีที่หาร 400 ลงตัว จึงกำหนดให้เดือนกุมภาพันธ์มี 29 วัน และถือว่าเป็นปีอธิกสุรทิน การคำนวณจึงต้องใช้กฎเกณฑ์ดังกล่าว แต่โปรแกรมประยุกต์จำนวนมากอาจคิดว่าปี 2000 เหมือนปี 1900 จึงให้เดือนกุมภาพันธ์มี 28 วัน ซึ่งเป็นผลทำให้การคิดคำนวณอย่างอื่นผิดพลาดตามมา

การเปลี่ยนแปลงหมายเลขโทรศัพท์

ในอดีตประเทศไทยเปลี่ยนหมายเลขโทรศัพท์จาก 6 หลัก เป็น 7 หลัก ในโซนกลาง และมีรหัสหมายเลขโซนเป็น 02, 034 ซึ่งอาจจะเรียกได้ว่าใช้สามหลัก รหัสโทรศัพท์ประเทศไทยจึงมี 10 หลัก การใช้ซอฟต์แวร์จึงเตรียมหลักโทรศัพท์ไว้ 10 หลักสูงสุด

ในสหรัฐอเมริกา ในประเทศจีน หรือประเทศที่มีประชาชนหลายร้อยล้านคนเช่นอินเดีย หากพัฒนาการให้ระบบโทรศัพท์ก้าวหน้าขึ้นมีการใช้โทรศัพท์แพร่หลาย การใช้รหัสพื้นที่ 3 หลักเช่นปัจจุบัน และรหัสโทรศัพท์ 7 หลักจะไม่พอเพียง กล่าวกันว่าภายในปี ค.ศ. 2015 ประเทศสหรัฐอเมริกาจะพบกับปัญหาการเปลี่ยนรหัสหมายเลขโทรศัพท์ครั้งใหญ่ ทั้งนี้เพราะระบบตัวเลข 10 หลักไม่พอเพียง

ในอดีตที่เคยเปลี่ยนเช่นในประเทศไทย เปลี่ยนจาก 6 หลัก เป็น 7 หลัก ระบบชุมสายเป็นระบบกลไก แต่ปัจจุบันทุกอย่างเป็นคอมพิวเตอร์ การเปลี่ยนหมายเลขทำได้ง่าย แต่ผลกระทบทางด้านซอฟต์แวร์ที่มีในระบบธุรกิจจะมีมากมายมหาศาล เพราะฟิลด์ข้อมูลที่เก็บไว้ไม่พอเพียง เหมือนเช่นปัญหาปี 2000

ยิ่งในอนาคตสังคมโลกเป็นสังคมโลภาภิวัตน์ ระบบโทรศัพท์อาจเป็นระบบเชื่อมโยงข้ามถึงกันได้ การใช้หมายเลขโทรศัพท์จึงต้องรองรับระบบที่เป็นแบบโลกาภิวัตน์ กล่าวกันว่า ทั้งโลกจะมีประชากรประมาณ 6,000 ล้านคน และมีแนวโน้มที่จะเพิ่มเป็น 10,000 ล้านคน ระบบเลขหมายโทรศัพท์ในอนาคตที่รองรับคนทั่วโลกได้ต้องเป็นระบบที่มีเป็นจำนวนทริลเลียน (trillion) หมายเลขนั่นคือระบบหมายเลขต้องใช้รวมกันถึง 12 หลัก

ปัญหาบนเครื่องยูนิกซ์และ C

ยูนิกซ์เป็นโอเอสที่พัฒนามาในช่วงทศวรรษที่ 1970 อุปกรณ์พัฒนามาบนพื้นฐานของภาษาซีที่ใช้กัน และนำมาเป็นตัวสำหรับพัฒนาระบบ ในขณะที่พัฒนายูนิกซ์ก็ได้มีการกำหนดข้อมูลวันที่ที่เก็บไว้ในรีจิสเตอร์ขนาด 4 ไบต์ (32 บิต) โดยเป็นรีจิสเตอร์เก็บเวลาเป็นวินาที โดยเริ่มจากวันที่ 1 มกราคม 1970 เป็นต้นมา ตัวเลขนี้มีค่าได้สูงสุดเต็มรีจิสเตอร์ที่ใช้คือ 2147483647 วินาที ซึ่งถ้าแปลกลับมาแล้ว ก็จะเป็นวันที่ 19 มกราคม 2038 เวลา 3:14:07

เมื่อเป็นเช่นนี้ ผู้ใช้ C หรือยูนิกซ์ที่นำฟังก์ชันวันที่ไปใช้จะมีข้อกำหนดอยู่ที่วันที่ 19 มกราคม 2038 และจะกลับมาเริ่มต้นใหม่ ดังนั้นในไลบรารีต่าง ๆ ของซีจะสร้างความสับสนและมีปัญหากับผู้ใช้ถ้าใช้วันที่เกินกว่ากรอบเวลานี้

การสร้างสไลด์วินโดว์

ในการแก้ปัญหาปี 2000 ระดับหนึ่งคือ การสร้างกรอบวันที่ใหม่ หรือที่เรียกว่าวินโดว์ กรอบวันที่นี้ยังคงใช้ตัวเลขสองหลักแทนปี ค.ศ. แต่มีความหมายใหม่ เช่น ในเอ็กเซลของไมโครซอฟต์เวอร์ชัน 5 สร้างกรอบวันที่

ไป 20 ปี คือ หากป้อน 00–19 จะหมายถึง 2000–2019 แต่หากป้อน 20–99 จะหมายถึง 1920–9999 แต่ในผลิตภัณฑ์ของไมโครซอฟต์ใหม่นี้สร้างกรอบ 30 ปี คือ จาก 00–29 จะหมายถึง 2000–2029

อย่างไรก็ดี การสร้างกรอบวินโดว์นี้มีข้อจำกัด กล่าวคือ การใช้กรอบวันที่ที่แตกต่างกัน และยังมีปัญหาในเรื่องเวลา หากเมื่อเวลาเข้าใกล้ก็จะมีปัญหาเหมือนปี 2000 อีก การสร้างกรอบวินโดว์จึงเป็นเรื่องที่สร้างปัญหาต่อเนื่องไม่จบสิ้น

ตัวอย่างปัญหาที่กระทบกับซอฟต์แวร์ยังมีอีก และจะยังคงเกิดขึ้นพบเห็นได้อีกมาก เพราะเรามีการพัฒนาซอฟต์แวร์กันมากขึ้น มีการใช้งานและมีการเปลี่ยนแปลงตามวิถีของชีวิตมนุษย์ การกระทบจึงเกิดขึ้น ตั้งแต่ระบบที่เป็นระบบเล็ก ๆ ในองค์กร เกิดกับองค์กรขนาดใหญ่ระดับประเทศและระดับโลก

เราจะแก้ปัญหาเหล่านี้ได้อย่างไร

การแก้ปัญหาที่เกิดขึ้นกับซอฟต์แวร์จึงอยู่ที่ว่า จะแก้ที่ปลายเหตุหรือต้นเหตุ การแก้ที่ปลายเหตุหมายถึง เมื่อพบเหตุการณ์ก็หาวิธีการแก้ไขและดำเนินการกับซอฟต์แวร์นั้น ๆ ซึ่งก็เป็นเรื่องที่ยุ่งยากมาก แต่การแก้ที่ต้นเหตุก็เป็นเรื่องที่เตรียมการหรือรับรู้ว่าจะเกิดปัญหา จึงได้พยายามป้องกันไม่ให้เกิดปัญหานั้น

การแก้ปัญหาที่ปลายเหตุคงต้องการสิ่งต่าง ๆ หลายอย่างเข้าช่วย เช่น

- หาวิธีการที่มีประสิทธิภาพในการค้นหาปัญหาและตรวจสอบ
- ต้องหาเครื่องมือที่จะใช้ในการปรับเปลี่ยนข้อมูลหรือโปรแกรมที่รวดเร็ว
- หาวิธีในการทดสอบโปรแกรม
- กำหนดรูปแบบมาตรฐาน

สังเกตได้ว่า การแก้ปัญหาที่ปลายเหตุเป็นการแก้ปัญหาที่ยากต่อการแก้ไขได้หมด เพราะปัญหาที่สั่งสมมานานและยากจะค้นหาได้ว่าอยู่ที่ใด การแก้ปัญหาปี 2000 ก็เป็นตัวอย่างหนึ่ง

การแก้ปัญหาที่ต้นเหตุคือการป้องกันไม่ให้เกิดหรือรู้ปัญหาแล้วแต่ก็ได้พยายามป้องกัน ในกรณีนี้อาจช่วยได้โดยเฉพาะถ้ามีการวางแผนการและกำหนดให้ผู้พัฒนาซอฟต์แวร์ดำเนินการอย่างเป็นระบบ

ลองนึกดูว่าถ้าทุกคนใช้มาตรฐานกลางในเรื่องการกำหนดวันที่เหมือนกันหมด โดยเฉพาะเน้นให้ใช้ตามมาตรฐาน ISO ดังตัวอย่างเรื่องฟอร์แมตวันที่–เวลา ตาม ISO 8601 ซึ่งนำออกมาใช้ตั้งแต่ปี ค.ศ. 1988 ปัญหาในเรื่อง 516/98 ก็คงไม่สับสนว่าเป็นเดือน 5 วันที่ 6 หรือเป็นเดือน 6 วันที่ 5 ตามมาตรฐาน ISO บอกให้ใช้ yyyy–mm–dd แต่สภาพความเคยชินในประเทศต่าง ๆ จึงไม่ยอมใช้มาตรฐานกลางนี้

รูปแบบที่ใช้งานกัน แยกตามประเทศ ได้รูปแบบแตกต่างกันมาก เช่น

yyyy–mm–dd เป็นมาตรฐาน ISO

mm–dd–yyyy ใช้ในสหรัฐอเมริกา

dd–mm–yy ใช้กันทั่วไป กรณีปี 2 หลักของยุโรป

dd–mm–yyyy ใช้กันทั่วไป กรณีปี 4 หลักของยุโรป

mm–dd–yy ใช้กรณีของสหรัฐอเมริกากรณีปี 2 หลัก

yyyy–ddd ใช้แบบจูเลียนและปีแบบสองหลัก

dddddddd ใช้ตามหลักดาราศาสตร์เริ่มต้นจากมกราคม 1,4713BL

ดังนั้นปัญหาในเรื่องวันที่แม้จะผ่านปี 2000 ไปแล้วก็ยังมีปัญหาอยู่ เพราะทั่วโลกใช้รูปแบบข้อมูลไม่เหมือนกัน และยากที่จะบอกให้แต่ละประเทศมายอมรับแบบ ISO

ความคิดจึงมีการเสนอรูปแบบที่ใช้ x เป็นตัวเลขอีกหนึ่งตัวเลขนำหน้า เพื่อบอกรูปแบบ โดยที่

ค่าของ x	รูปแบบข้อมูล
1	yyyy-mm-dd
2	mm-dd-yyy
3	dd-mm-yy
4	dd-mm yyyy
5	mm-dd-yy
6	yy-ddd
7	yyyy-ddd
8	dddddddd

ซอฟต์แวร์ที่รับข้อมูลวันที่มาจึงเข้าใจได้ว่ารูปแบบที่ใช้เป็นรหัสตามความต้องการของประเทศใด

การแก้ปัญหาจึงอยู่ที่ความพยายาม และหาทางเรียนรู้ให้ได้มากที่สุด เข้าใจรูปแบบปัญหา และหาทางป้องกันไม่ให้เกิดปัญหา หรือมีแนวทางที่ใช้ในการแก้ปัญหาเมื่อมีความจำเป็นต้องเปลี่ยนรูปแบบข้อมูลที่ใช้

เอกสารอ้างอิง

1. Capeis Jones "Bad Days for Software", IEEE Spectrum, Vol. 35, No. 9, Setpz 1998, pp.47-52